DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION	ATTORNEY DOCKET NO. 10008270-1

As a below named inventor, I hereby declare that:

My residence/post office address and citizenship are as stated below next to my name;

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled:

MODELING TOOL FOR ELECTRONIC SERVICES AND ASSOCIATED METHODS

the specification of which is attached hereto unless the following box is checked:

( ) was filed on _____ as US Application No. or PCT International Application

Number _____ and was amended on _____ (if applicable).

I hereby state that I have reviewed and understood the contents of the above-identified specification, including the claims, as amended by any amendment(s) referred to above. I acknowledge the duty to disclose all information which is material to patentability as defined in 37 CFR 1.56.

Foreign Application(s) and/or Claim of Foreign Priority

I hereby claim foreign priority benefits under Title 35, United States Code Section 119 of any foreign application(s) for patent or inventor(s) certificate listed below and have also identified below any foreign application for patent or inventor(s) certificate having a filing date before that of the application on which priority is claimed:

COUNTRY	APPLICATION NUMBER	DATE FILED	PRIORITY CLAIMED UNDER 35 U.S.C. 119	
			YES:	NO:
			YES:	NO:

Provisional Application

I hereby claim the benefit under Title 35, United States Code Section 119(e) of any United States provisional application(s) listed below:

APPLICATION NUMBER	FILING DATE

U.S. Priority Claim

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code Section 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, Section 1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

APPLICATION NUMBER	FILING DATE	STATUS (patented/pending/abandoned)

POWER OF ATTORNEY:

As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith:

Customer Number | 022879

Place Customer Number Bar Code Label here

Send Correspondence to:
HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

Direct Telephone Calls To:

Thomas X. Li
650-857-5972

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Inventor: Fabio Casàti   Citizenship: Italy

Residence: Palo Alto, California

Post Office Address: Hewlett-Packard Co., Legal, 1501 Page Mill Rd., Palo Alto CA 94304

Inventor's Signature   Date July 20, 2001

Full Name of # 2 joint inventor: Ming-Chien Shan   Citizenship: USA

Residence: Saratoga, California

Post Office Address: Hewlett-Packard, Legal Dept., 1501 Page Mill Rd., Palo Alto CA 94304

Inventor's Signature _____   Date 7/20/2001

Full Name of # 3 joint inventor: Mehmet Sayal   Citizenship: Turkey

Residence: Sunnyvale, California

Post Office Address: Hewlett-Packard, Legal Dept., 1501 Page Mill Rd., Palo Alto CA 94304
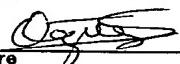
Inventor's Signature _____   Date 7/20/2001

Full Name of # 4 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 5 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 6 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 7 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 8 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Rev 08/01 (DocPwr)   (Use Page Two For Additional Inventor(s) Signature(s))

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE EXAMPLE SYSTEM "CSDL_Example.dtd">

<Composite-Service>

    <Meta-Model Name="Food Delivery" Version="A.01.00">

        <!-- Case packet data. These are the definitions of variables used by this composite service. -->
        <Composite-Service-Data>
            <Data-Item-Declaration Name="RestaurantName" Data-Type="STRING">
                <Value>%RestaurantName</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="OrderNumber" Data-Type="INTEGER">
                <Value>%OrderNumber</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="Order" Data-Type="STRING">
                <Value>%Order</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="CustomerCCN" Data-Type="STRING">
                <Value>%CustomerCCN</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="DeliveryTime" Data-Type="STRING">
                <Value>%DeliveryTime</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="DeliveryAddress" Data-Type="STRING">
                <Value>%DeliveryAddress</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="CustomerName" Data-Type="STRING">
                <Value>%CustomerName</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="Confirmation" Data-Type="INTEGER">
                <Value>%Confirmation</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="RestaurantCCN" Data-Type="STRING">
                <Value>%RestaurantCCN</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="CCExpiry" Data-Type="STRING">
                <Value>%CCExpiry</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="BillAmount" Data-Type="REAL">
                <Value>%CustomerName</Value>
            </Data-Item-Declaration>

            <!-- The following two variables are used in Search-Recipe. Search-Recipe is used in E-speak
                 to look up for services. -->
            <Data-Item-Declaration Name="mycondition" Data-Type="STRING">
                <Value>%mycondition</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="mypreference" Data-Type="STRING">
                <Value>%mypreference</Value>
            </Data-Item-Declaration>
            <Data-Item-Declaration Name="localhost" Data-Type="STRING">
                <Value>%localhost</Value>
            </Data-Item-Declaration>

            <!-- Input variables -->
            <Input>
                <Data-Item Name="CustomerCCN" />
                <Data-Item Name="CCExpiry" />
                <Data-Item Name="Order" />
                <Data-Item Name="DeliveryTime" />
                <Data-Item Name="DeliveryAddress" />
                <Data-Item Name="mycondition" />
                <Data-Item Name="mypreference" />
                <Data-Item Name="localhost" />
            </Input>

            <!-- Output variables -->
            <Output>
                <Data-Item Name="Confirmation" />
```

```xml
    <Data-Item Name="RestaurantName" />
    <Data-Item Name="OrderNumber" />
    <Data-Item Name="BillAmount" />
</Output>

<!-- Local variables: used only within the composite service -->
<Local>
    <Data-Item Name="RestaurantCCN" />
    <Data-Item Name="CustomerName" />
</Local>

</Composite-Service-Data>
```

```xml
<!-- Service-Flow-Structure describes the service flow and consists of route nodes,
    method nodes, and arcs -->
<Service-Flow-Structure>

    <!-- If the user's credit card was confirmed, the service continues to with delivery
        schedule; otherwise the service aborts. The condition below checks the value
        of the Confirmation variable in order to figure out whether the credit card was
        confirmed. -->
    <Route-Node Name="Check Passed" Description="Is CCN confirmed" Type="XOR-SPLIT">
        <Rule>
            <Condition>Confirmation &lt;&gt; 0</Condition>
                <Action>Goto Node Check And Join</Action>
                <Alternative-Action>Goto Node Abort</Alternative-Action>
        </Rule>
    </Route-Node>

    <!-- This is a join node used for joining two separate branches of the service flow, after the
        user's credit card is confirmed and the restaurant is selected. -->
    <Route-Node Name="And Join" Description="Join node" Type="AND-JOIN">
        <Rule>
            <Condition>TRUE</Condition>
                <Action>Goto Node Wheel Delivery</Action>
        </Rule>
    </Route-Node>

    <!-- This service carries out the confirmation of user's credit card. It takes one output, namely
        the credit card number, and returns a confirmation number. If the credit check fails, then
        the return value equals to zero -->
    <Service-Node Name="Check Credit" Description="Confirm credit card number">
        <!-- Search-recipe is used for service lookup in E-speak. It is used in the composite service
            in order to lookup the individual E-speak services that constitute the composite service.
            There are three workflow variables embedded inside the Search-Recipe: localhost,
            mycondition, and mypreference. These variables are replaced by their current values
            by the Gateway during the execution of the composite service. The localhost variable
            contains the URI of the local host machine of the user who is using the composite service.
            The mycondition variable contains the search criteria. The mypreference variable contains
            the sorting criteria in case multiple candidate services are found by the search recipe. All
            those three variables are input to the composite service by the user. This search recipe
            returns the first service that satisfies the given condition and is ranked the first after sorting
            preference is applied. -->
        <Search-Recipe>
            <?xml version="1.0">
            <header xmlns="www.e-speak.net/Schema/E-speak.header.xsd">
                <communication>
                    <to>es://%localhost/WebAccess/FindService</to>
                    <from>es://%localhost/WAUser1</from>
                </communication>
            </header>
            <esquery xmlns="www.e-speak.net/Schema/E-speak.query.xsd">
                <from>src="es:%localhost"/>
                <result>$serviceinfo</result>
                <where>%mycondition</where>
                <preference>%mypreference</preference>
                <!-- Arbitration is fixed to only one cardinality since we want only one result -->
                <arbitration>
                    <operator>first</operator>
                    <cardinality>1</cardinality>
```

```
        </arbitration>
      </esquery>
    </Search-Recipe>

    <!-- The following method  confirms the user's credit card. -->
    <Method-Node Name="Check_CCN_Node" Description="Confirm CCN">
      <Method-Name>CheckCCN</Method-Name>
        <Method-Input>
          <Value>%CustomerCCN</Value>
        </Method-Input>
        <Method-Output>
          <Var-Mapping FLOW-VAR="Confirmation" />
        </Method-Output>
    </Method-Node>
    <!-- Example certificate: client's certificate is used -->
    <Certificate Type="USER" />
    <!-- Skipped the optional Service-Exception-Handling element -->
</Service-Node>


<!-- This service performs the restaurant selection. It takes two inputs: user order and
    delivery time. It returns the name of the selected restaurant. It consists of only one method
    which performs the restaurant selection. -->
<Service-Node Name="Restaurant Selection" Description="Select a restaurant">
    <Search-Recipe>
      <!-- Search recipe is the same as that of Check Credit node -->
    </Search-Recipe>
    <Method-Node Name="Select_Restaurant_Node" Description="Select Restaurant">
      <Method-Name>SelectRestaurant</Method-Name>
        <Method-Input>
            <Input-Var-Mapping FLOW-VAR="Order" Method-Var="p0">
            <Input-Var-Mapping FLOW-VAR="DeliveryTimer" Method-Var="p1">

</Method-Input>
          <Method-Output>
            <Var-Mapping FLOW-VAR="RestaurantName" />
          </Method-Output>
        </Method-Node>
      <!-- Skipped the optional Certificate and Service-Exception-Handling elements -->
    </Service-Node>


<!-- The Wheel Delivery service consists of two methods. Therefore, it contains a
    Method-Flow-Structure. The first method orders the food and the second one
    Charges service fee to the restaurant. -->
<Service-Node Name="Wheel Delivery" Description="Order Food and Get Payment">
    <Search-Recipe>
      <!-- Search recipe is the same as that of Check Credit node -->
    </Search-Recipe>
    <Method-Flow-Structure>
      <!-- This method orders the food from the selected restaurant. It takes four inputs: restaurant
          name, order, delivery time and address. An E-speak service method returns only one
          output. However, two output values are expected from this method: order number and
          bill amount (how much to charge the user). The Var-Mapping tags also include
          Conversion-Rule attributes which describe how to extract two expected outputs from the
          method's single output value. -->
      <Method-Node Name="Order_Food" Description="Order Food">
        <Method-Name>OrderFood</Method-Name>
        <Method-Input>
          <Value>%RestaurantName</Value>
          <Value>%Order</Value>
          <Value>%DeliveryTime</Value>
          <Value>%DeliveryAddress</Value>
        </Method-Input>
        <!-- XQL is used for extracting two output values from a single method output in the following
            Var-Mapping tags. -->
        <Method-Output>
          <Var-Mapping FLOW-VAR="OrderNumber" Conversion-Rule="OrderNumber/Orderresult[0]
              Rule-Type="XQL" />
          <Var-Mapping FLOW-VAR="BillAmount" Conversion-Rule="BillAmount/Orderresult[1]
              Rule-Type="XQL" />
        </Method-Output>
```

```
    </Method-Node>
    <Method-Node Name="Get_Payment" Description="Get payment from the restaurant">
        <!-- This method charges a service fee to the selected restaurant. It takes two inputs:
            order number and restaurant's credit card number (or account number). It does not
            return any output since a confirmation of the restaurant's account or credit card
            is not required. -->
        <Method-Name>GetPayment</Method-Name>
        <Method-Input>
            <Value>%OrderNumber</Value>
            <Value>%RestaurantCCN</Value>
        </Method-Input>
        <!-- No method output exists for this method. Skipping Method-Output element -->
    </Method-Node>
    <!-- The following tags describe the arcs inside the service node's Method-Flow-Structure -->
    <Arc Type="Forward" Source="Start" Destination="Order_Food" />
    <Arc Type="Forward" Source="Order_Food" Destination="Get_Payment" />
    <Arc Type="Forward" Source="Get_Payment" Destination="Stop" />
    </Method-Flow-Structure>
    <!-- Skipped the optional Certificate and Service-Exception-Handling elements -->
</Service-Node>

<!-- This service charges the bill amount, which was returned from a previous method call,
    to the user's credit card. The service consists of a single method which takes two inputs:
    bill amount and the user's credit card number. No output is required from that method. -->
<Service-Node Name="Credit Card" Description="Charge the customer credit card">
    <Search-Recipe>
        <!-- Search recipe is the same as that of Check Credit node -->
    </Search-Recipe>
    <Method-Node Name="Credit_Card_Node" Description="Charge credit card">
        <Method-Name>CreditCard</Method-Name>
        <Method-Input>
            <Value>%BillAmount</Value>
            <Value>%CustomerCCN</Value>
        </Method-Input>
        <!-- No method output exists for this method. Skipping Method-Output element -->
    </Method-Node>
    <!-- Skipped the optional Certificate and Service-Exception-Handling elements -->
</Service-Node>

<!--The following arcs describe the service flow among the routing and service nodes of
    the composite service. Note that no start, stop and complete nodes are defined previously
    within the Service-Flow-Structure. Virtual references to Start, Stop and Complete nodes
    are used only in the arcs in order to indicate the beginning and end points of the composite
    service. Those Start, Stop and Complete nodes do not involve any method invocations
    in E-speak. Therefore, they do not have to be explicitly declared prior to their use in
    the arcs. -->
<!-- Arcs of the composite service -->
<Arc Type="Forward" Source="Start" Destination="Check Credit" />
<Arc Type="Forward" Source="Start" Destination="Restaurant Selection" />
<Arc Type="Forward" Source="Check Credit" Destination="Check Passed" />
<Arc Type="Forward" Source="Check Passed" Destination="Stop" />
<Arc Type="Forward" Source="Check Passed" Destination="And Join" />
<Arc Type="Forward" Source="Restaurant Selection" Destination="And Join" />
<Arc Type="Forward" Source="And Join" Destination="Wheel Delivery" />
<Arc Type="Forward" Source="Wheel Delivery" Destination="Credit Card" />
<Arc Type="Forward" Source="Credit Card" Destination="Complete" />

</Service-Flow-Structure>

</Meta-Model>

</Composite-Service>
```